**Boundary Traversal of Binary Tree**

bool isLeaf(TreeNode<int>\* root) {

  return !root -> left && !root -> right;

}

void addLeftBoundary(TreeNode<int>\* root, vector < int > & res) {

  TreeNode<int>\* cur = root -> left;

  while (cur) {

    if (!isLeaf(cur)) res.push\_back(cur -> data);

    if (cur -> left) cur = cur -> left;

    else cur = cur -> right;

  }

}

void addRightBoundary(TreeNode<int>\* root, vector < int > & res) {

 TreeNode<int>\* cur = root -> right;

  vector < int > tmp;

  while (cur) {

    if (!isLeaf(cur)) tmp.push\_back(cur -> data);

    if (cur -> right) cur = cur -> right;

    else cur = cur -> left;

  }

  for (int i = tmp.size() - 1; i >= 0; --i) {

    res.push\_back(tmp[i]);

  }

}

void addLeaves(TreeNode<int>\* root, vector < int > & res) {

  if (isLeaf(root)) {

    res.push\_back(root -> data);

    return;

  }

  if (root -> left) addLeaves(root -> left, res);

  if (root -> right) addLeaves(root -> right, res);

}

vector<int> traverseBoundary(TreeNode<int>\* root){

  vector < int > res;

  if (!root) return res;

  if (!isLeaf(root)) res.push\_back(root -> data);

  addLeftBoundary(root, res);

  // add leaf nodes

  addLeaves(root, res);

  addRightBoundary(root, res);

  return res;

}